Atto[illegible]ocket No.: 10559/096001/P7615

APPLICATION

FOR

UNITED STATES LETTERS PATENT

[illegible]

TITLE:	REAL-TIME PERFORMANCE ASSESSMENT OF LARGE AREA NETWORK USER EXPERIENCE
APPLICANT:	CYNTHIA L. BICKERSTAFF, MARK A. VANANTWERP, WILLIAM W. LOVE, STACY P. PURCELL, LAWRENCE J. WIDMER AND JEFFREY C. SEDAYAO

CERTIFICATE OF MAILING BY EXPRESS MAIL

Express Mail Label No. EL528179594US

I hereby certify that this correspondence is being deposited with the United States Postal Service as Express Mail Post Office to Addressee with sufficient postage on the date indicated below and is addressed to the Assistant Commissioner for Patents, Washington, D.C. 20231.

December 23, 1999
Date of Deposit

[illegible]
Signature

R. Donovan
Typed or Printed Name of Person Signing Certificate

REAL-TIME PERFORMANCE ASSESSMENT OF LARGE AREA NETWORK USER EXPERIENCE

TECHNICAL FIELD

This invention relates to measuring and modifying the performance of a network, and more particularly to a method, system, and computer program for real-time measurement and modification of the performance of communications on a large area network, such as the Internet, based upon actual user experience.

BACKGROUND

FIG. 1 shows a typical configuration of a large area network, such as the Internet. Multiple geographically dispersed user client systems 100 are connected through user Internet Service Providers (ISPs) 102 to the network "cloud" 104 comprising the Internet backbone communication transport systems. Requests for information or services by a client application program executing on a user client system 100 may be routed through multiple server ISPs 106 through a router 108 to a web server 110. The web server 110 retrieves or generates the requested information or provides the requested service, and communicates a response to the requesting client application.

Various attempts have been made to actively assess the performance (*e.g.*, response time, transmission problems, *etc.*)

of Internet connections. Some current methods use active measurement of various network or server components that are dedicated to performance measurement. For example, conventional Internet Control Message Protocol (ICMP) "ping" and "traceroute" commands can be used to measure the performance of the network connections between a client terminal and a server. However, these commands are frequently transmitted with a different (often lower) priority than the protocols used by applications run by users for "web surfing". Accordingly, inaccurate (*i.e.*, false positive) measurements are common.

Attempts have been made to actively measure Internet connections using the same protocols used by end-user applications, such as the HyperText Transfer Protocol (HTTP) "GET" command. These approaches typically use computer programs (sometimes known as "hosts", "agents", or "beacons") residing on measurement instrumentation capable of communicating with Internet protocols. However, such computer programs are limited to assessing network paths only from the specific network nodes on which they are executing. Further these techniques inject traffic into sometimes overburdened Internet, WAN, or LAN infrastructures, causing the measurement process to change the characteristics being measured. Additionally, these techniques are relatively expensive to implement.

A further problem of all of these active or injected measurement approaches is that they generate non-value added communication traffic for both local and large area network infrastructures.

SUMMARY

In one aspect, the invention includes a method, system, and computer program for real-time measurement of the performance of communications on a large area network between a selected server and a plurality of users, based upon actual user experience, including: accessing a server log having records of actual user access to the selected server; aggregating records from the server log into a plurality of aggregate slots, each having at least one time bin, based on an aggregation method; performing at least one statistical analysis of each time bin of each aggregate slot; and outputting the results of such statistical analysis as an indication of actual server usage by users.

The details of one or more embodiments of the invention are set forth in the accompanying drawings and the description below. Other features, objects, and advantages of the invention will be apparent from the description and drawings, and from the claims.

DESCRIPTION OF DRAWINGS

FIG. 1 shows a typical configuration of a large area network, such as the Internet.

FIG. 2 is a process flow diagram showing one embodiment of the invention.

FIG. 3 is a flowchart for a process comparing information from a Classless Inter-Domain Routing (CIDR) block database and an Internet Protocol (IP) address input in order to convert the IP address to geographic or source information according to one embodiment of the invention.

FIG. 4 is a flowchart showing an embodiment for modifying traffic paths through a router to the Internet.

Like reference numbers and designations in the various drawings indicate like elements.

DETAILED DESCRIPTION

Embodiments of the invention are directed to a method, system, and computer program for real-time measurement and modification of the performance of communications on large area networks, such as the Internet, based upon actual user experience. One embodiment performs a statistical analysis of access logs that record actual server usage by users. Based on such analysis, routing of communications over the network can be modified to improve overall communications performance. Embodiments may also output results indicative of overall communications performance and of server applications that interact poorly or especially well with network conditions, thus providing direction to application development efforts.

More particularly, one embodiment of the invention creates correlation assessments of performance related measurements against the geographical location of and/or routes taken by client applications. A route is determined by aggregating client Internet Protocol (IP) addresses according to Classless Inter-Domain Routing (CIDR) blocks or route advertisements available in a conventional fashion by querying a router or router server for such advertisements.

The results of these analyses define which geographical location or route may be performing better or worse than a comparative geographical location or route. Based on such comparisons, active steps may be taken to modify routing of network traffic to increase overall client-server performance.

In addition, each web server running a set of applications can be compared with every other server running a set of applications within the same domain. Such a comparison can detect differences in configuration of the servers, and permits identification of servers that are providing poor performance to users. Based on such comparisons, active steps may be taken to modify the configurations of dissimilar servers to match the performance of other servers within a group of evaluated servers.

An advantage of using such log file analysis over active measurements for detecting performance over a large area network such as the Internet is that historical records of an end user's experience can be mined for objective quantitative information and compared to the experience of other end users collected at or near the same time. This allows for identifying the root cause of performance problems. Since actual user experience is assessed, the limitation of a few, expensive sampling locations of beacons or agents is eliminated. The integrity of the analysis for any individual web site is limited by the popularity of a web site residing on a web server. However, an enterprise hosting multiple web sites can alleviate this limitation by aggregating across multiple web sites for the same end-user population. Tuning the performance of a web site to those users already using it enhances current users' experience.

One embodiment of the invention creates Pareto analyses of different applications running on a server where applications taking longer than a configurable time interval for greater percentages of the use of the application by end users are sorted in order from "most often" to "least often" exceeding the interval. Based on such an analysis, allocation of application developer resources to poorest performing applications can be made to improve the application and improve the end user experience.

)

Statistical Analysis Process Flow

FIG. 2 is a process flow diagram showing an embodiment of the invention adapted for use with the Internet. One input to the process is a server log file 200 that is maintained on a web server 110 (see FIG. 1) or a mounted file system in conventional fashion. In the illustrated embodiment, the server log file 200 is configured according to World Wide Web Consortium (W3C) standards, and includes a log file 202 for each web server 110 being monitored within a server group. A typical server log file 200 configured to such standards records all user accesses of every element of a web page. A typical logical organization for the server log file 200 is a table having columns for every recorded data item, and rows for each access event. In particular, a server log file 200 that is most suitable for use with the invention records the following data for each user access of every element of a monitored web page: a time stamp indicating when a record was created in the server log file 200; client IP (c_ip) address; bytes transmitted from client application to server (cs_bytes); bytes transmitted from server to client application (sc_bytes); the time taken to complete a two-way transmission of bytes between a client and a server (time_taken); status codes documenting an action for each web page element (status_code); each URL (uniform resource locator) requested by a client browser (uniform resource identifier stem, or uri_stem); the type of browser used by the end-user (user-agent); and each URL referring to the uri_stem (uniform resource identifier referrer, or uri_referrer). Other information may also be included in each record, as desired or in order to comply with Internet standards.

In the illustrated embodiment, any particular server log file 200 is closed to new data entries before commencement of any statistical analysis. A new log file may be opened in known fashion to continue to record user access while the closed log file 200 is analyzed. Embodiments of the invention may also use log file entries written from a server directly to a flat file or database.

Sub A1 7 Process parameters are defined in a process settings step 202. In this step, an analyst either selects an aggregation method (e.g., "aggregate by log-file column", or "aggregate by client IP address"), optional filtration parameters, and an aggregation bin time increment, or these parameters set by reference to default (*i.e.*, pre-established) settings.

Sub A2 7 If filtration parameters are set in the process settings step 202, the data in the server log file 200 is filtered to remove records that are not to be counted in further statistical analyses. For example, such records may be from

Sub A2 7

non-customer sources, such as a beacon or agent, and thus do not reflect actual user accesses to the web server 110. In the illustrated embodiment, an agent ID field within a conventional W3C compliant server log file is used to filter out undesirable records. However, any desired record field may be used to perform a selected filtration. In the illustrated embodiment, filtering is implemented as a string matching function that compares a filter string to any character string or substring in any of the log file fields. Other types of filtering may by employed, such as by comparing the client IP (c_ip) address against a "lookup" table of addresses to include or exclude.

The selected log file records are then processed in an aggregation step 208 using the aggregation method defined in the process settings step 202. Typical aggregation methods are an "aggregate by log-file column" method 210 (*e.g.*, AS-path, country, region, *etc.*) or an "aggregate by client IP address" method 212. The aggregation method creates entries within an aggregation table 216 having multiple aggregate slots 218 each generally having multiple time bins 220.

For example, the log-file column aggregation method 210 reads a defined record column (or "field") data value and time stamp for each selected log file record and assigns that record to a corresponding time bin 220 within the appropriate aggregate slot 218. Thus, records accumulated over a 24-hour period and corresponding to a first defined column data value can be assigned to 24 1-hour time bins 220 in a first aggregate slot 218, while records corresponding to a second defined column data value are assigned to 24 1-hour time bins 220 in a second aggregate slot 218.

If the "aggregate by client IP address" method 212 is selected in the process settings step 202, it is generally desirable to convert the raw IP address of a user client system 100 accessing the web server 110 to a geographic location or specific source (*e.g.*, country, region, company, and/or ISP). In the illustrated embodiment, this is accomplished by supplying the client IP (c_ip) address from each record to an IP Lookup function 214, which returns geographic location or specific source information associated with that address. One implementation of the IP Lookup function 214 is described in detail below.

Once the aggregation table 216 has been created, the log file entries within each time bin 220' may be subjected to a statistical analysis process 222. This process applies a variety of statistical analysis algorithms 224 to derive information on server usage and statistical significance of such information based on the actual user access records. Collections of multiples of such time-bins 220' can also be assembled in chronological order to determine trends for each of the statistical measures. In the illustrated embodiment, the basic rate and count information computed are: byte-density (computed as sc_bytes + cs_bytes); transfer-rate (computed as byte-density divided by time taken); URL-count (total number of log entries); error-fraction (the fraction of all log entries having errors); cache-fraction (the fraction of all log entries having cached URL's as determined by response code); and unique-IP-address-count (the number of unique IP addresses among all log entries). Once the basic rate and count information is computed, distribution statistics may be computed for some or all of such basic information. In particular, in the illustrated embodiment, distribution statistics, such as quartiles, interquartile range (IQR), and median, are computed in known fashion for the byte-density and transfer-rate statistics.

The results of the statistical analysis process 222 can generate output 226 in several forms. The raw data from the statistical analysis process 222 can be output directly. Trend information 228 can be output (*e.g.*, in table or graphical form) to show the trends of time bins by aggregate slot or item. Various comparison tests (*e.g.*, out of range, over threshold, percentage change, *etc.*) can be applied to the basic rate and count information as well as the distribution statistics to trigger an event notification 230 (*e.g.*, notice to a network administrator) if any selected statistical value is abnormal. Further, the statistical information for multiple time bins and/or aggregate items can be input to various comparison tools 232 for troubleshooting. For example, the IQR statistics for the byte-density for two servers within a domain can be compared graphically for visual assessment by a network administrator. Generation of such trend displays, event notifications, and comparisons is well-known in the art.

Thus, the illustrated embodiment of the invention can create correlation assessments of performance related measurements against the geographical location and/or route traversed during use of a network application by an end-user. In particular, transfer-rate and error-fraction measurements can be correlated to at least the following parameters: geographical location of c_ip addresses; ISP for c_ip addresses; net block or route of c_ip addresses; and applications requested (uri_stem) or previously requested (uri_referrer) by client applications or users from the web server 110. The results of these analyses define which geographical location, ISP, net block, route, or application may be performing better or worse than a comparative geographical location, ISP, net block, route, or application.

The validity of the correlations is ensured by performing statistical validity checks between applications and servers, such as by ensuring similarity or sufficiency of certain of the computer distribution statistics, in known fashion. The byte-density, URL-count, and unique-IP-address-count parameters are used to ensure valid correlations. For example, since the common TCP/IP protocol (the protocol used over the Internet) changes its performance based on the number of packets transmitted (through congestion control and "slow start" mechanisms), requiring a similar value for the byte-density parameter ensures that differences between servers or services of different applications are due to other interesting parameters (such as the geographical location of c_ip addresses, ISP for c_ip addresses; *etc.*) instead of resulting from artifacts (*e.g.*, large byte transfers generated by the TCP/IP protocol itself). The combination of the URL-count and the unique-IP-address-count parameters represent the sample size of the analysis space. Since each unique IP address essentially represents a different end-to-end communication path, the unique-IP-address-count measures the diversity of the network space being measured. Requiring that the URL-count and the unique-IP-address-count parameters exceed a selected threshold helps ensures that the correlations described above are valid.

If the correlations described above indicate a problem, actions may be undertaken to rectify the problem. These actions may include: selecting a better exit path from a multi-homed (*i.e.*, having multiple ISPs) data center (described in greater detail below); notifying a network administrator to repair a server which is performing below the level of ostensibly identical servers; and indicating the need to re-write applications which are slow performing. For this later case, it may be that the applications perform well during local area network testing, but log file analysis in accordance with the invention may reveal an application specific sensitivity to actual Internet conditions.

IP Lookup function

FIG. 3 is a flowchart showing an embodiment for comparing information from a Classless Inter-Domain Routing (CIDR) block database 300 and an IP address input 304 in order to convert the IP address to geographic or source information. A CIDR block defines a subnet of a larger network. A CIDR address includes a standard 32-bit IP address and also information on how many bits are used for the network prefix. This addressing scheme allows for efficient allocation of IP addresses compared to prior standards. The CIDR addressing scheme also enables "route aggregation" in which a single high-level route entry can represent many lower-level routes in global routing tables.

In the illustrated embodiment for the Internet, the CIDR block database 300 is specially generated by querying (using conventional Internet query commands) regional Internet registries for CIDR blocks that have been assigned through such registries. The responses from the registries include

CIDR block address (used as the database key), country code, network name, network description, region (*i.e.*, sub-country geographical information, sometimes down to a street address), and date of the last update for each registry record.

During operations, the CIDR block database 300 may be read into memory organized as a 32-element array 302. Each array element 303 is a binary tree of CIDR block records selected with a unique subnet mask value. For example, array element "0" contains a binary tree of all CIDR block records whose subnet mask is "255.255.255.255" (*i.e.*, having a binary representation of 32 "1's"). Similarly, array element "1" contains a binary tree of all CIDR block records whose subnet mask is "255.255.255.254" (*i.e.*, having a binary representation of 31 "1's" followed by one "0" as the least significant bit). This pattern continues, such that array element "31" contains a binary tree of all CIDR block records whose subnet mask is "1.0.0.0" (*i.e.*, having a binary representation of one "1" followed by 31 "0's").

The subnet mask for each array element is used to mask a target IP address before searching the element's associated binary tree. The subnet mask can be computed from the CIDR block mask number as the binary complement of $2^{32\text{-MaskNumber}} - 1$. This configuration of CIDR blocks in the memory array 302 provides for most specific CIDR block/IP address matching.

In operation, a target c_ip address from a record in the server log 200 is used as input to the most specific CIDR block/IP address matching process (STEP 304). For each c_ip address, a counter *N* is set to "0", representing array element "0" (STEP 306). Using the subnet masking technique described above, the target c_ip address is masked with the array element's associated subnet mask (*e.g.*, all "1's" for array element "0"), and the corresponding array element's binary tree is then traversed to find a record match (STEP 308). In particular, the masked IP address component of each CIDR block for each record traversed in the N^{th} binary tree is compared against the masked target IP address.

If a match exists, then desired record fields from the corresponding CIDR block (e.*g*., country code, network name, network description, region, and/or date) are sent to output to be used for binning by the lookup requestor (STEP 310). Thus, the c_ip address is converted to geographical and source information.

If no match occurs (STEP 308), *N* is incremented and tested for being in the range 0-31 (STEP 312). If *N* is out of range, no match exists and is so indicated (STEP 314). Otherwise, the match process continues with the next array element through similar masking of the target c_ip address and traversal of the associated binary tree for the incremented value of *N*.

17

Active ISP Routing

Sub A³ FIG. 4 is a flowchart showing an embodiment for modifying traffic paths through a router to a large area network such as the Internet. After the statistical analysis describe above is performed, the results can be used to "tune" performance of a server system. In the illustrated embodiment, the exit route for communications from a web server 110 through the router 108 and all connected server ISPs 106 to the network 104 is determined for each c_ip address (STEP 400). This may be accomplished by querying the router 108 (or a route server), using conventional network control commands, for the routing table maintained by the router 108. The routing information may then be analyzed to determine which exit path has the highest performance (*e.g.*, highest transfer-rate for a particular destination).

Sub A⁴ Once a preferred exit route is determined, the routing of traffic may be biased towards that exit route (or, alternatively, away from the most poorly performing exit routes). For the Internet, this may be done using Border Gateway Protocol (BGP) mechanisms. BGP is commonly used as a router-to-router protocol between administrative domains. For example, in the illustrated embodiment, outgoing traffic is biased by modifying incoming routing update information using BGP path prepending or local preference mechanisms. Similarly, incoming traffic is biased by modifying outgoing routing

Sub A47

update information using BGP path prepending or community string mechanisms.

Implementation

The invention may be implemented in hardware or software, or a combination of both (*e.g.*, programmable logic arrays). Unless otherwise specified, the algorithms included as part of the invention are not inherently related to any particular computer or other apparatus. In particular, various general purpose machines may be used with programs written in accordance with the teachings herein, or it may be more convenient to construct more specialized apparatus to perform the required method steps. However, preferably, the invention is implemented in one or more computer programs executing on programmable systems each comprising at least one processor, at least one data storage system (including volatile and non-volatile memory and/or storage elements), at least one input device, and at least one output device. The program code is executed on the processors to perform the functions described above.

Each such program may be implemented in any desired computer language (including machine, assembly, or high level procedural, logical, or object oriented programming languages) to communicate with a computer system. In any case, the language may be a compiled or interpreted language.

Each such computer program may be stored on a storage media or device (*e.g.*, solid state, magnetic, or optical media) readable by a general or special purpose programmable computer, for configuring and operating the computer when the storage media or device is read by the computer to perform the procedures described herein. The inventive system may also be considered to be implemented as a computer-readable storage medium, configured with a computer program, where the storage medium so configured causes a computer to operate in a specific and predefined manner to perform the functions described herein.

A number of embodiments of the present invention have been described. Nevertheless, it will be understood that various modifications may be made without departing from the spirit and scope of the invention. Accordingly, other embodiments are within the scope of the following claims.